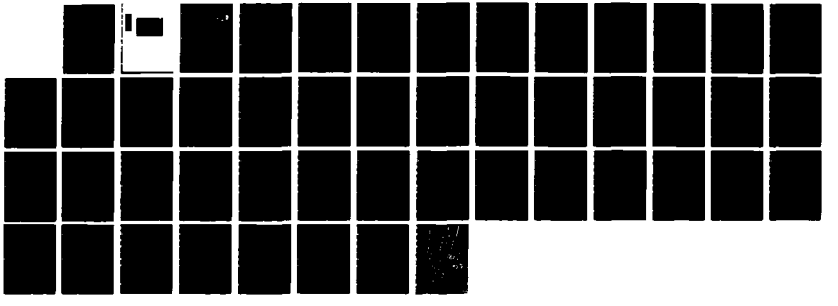
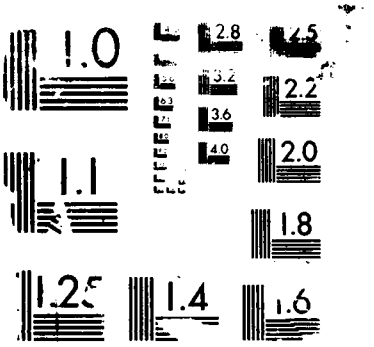
AD-A189 239 ONLINE AIDING FOR HUMAN-COMPUTER INTERFACES(U) MICHIGAN 1/1
UNIV ANN ARBOR CENTER FOR ERGONOMICS J ELKERTON
09 OCT 87 C4E-ONR-1 N00014-87-K-0740

UNCLASSIFIED F/G 23/2 NL





ONLINE AIDING FOR HUMAN-COMPUTER INTERFACES

Jay Elkerton
Center for Ergonomics
The University of Michigan

Technical Report C4E-ONR-1
October 9, 1987

ABSTRACT

Current research is surveyed on interfaces which aid the computer user

. The results of this review revealed that state-of-the-art knowledge in the design of these aiding interfaces is lacking. Designers of aiding interfaces are only provided qualitative design principles such as make the online help task oriented. As a result, many online aiding dialogues fall far short of the ultimate goal of helping users with their current problems, while also supporting continued skill acquisition at the computer interface. To address this problem, a task-analytic approach is presented which is based on the GOMS model (Card, Moran, & Newell, 1983) of human-computer interaction. This theoretical approach allows online aiding dialogues to be specified using the goals, operators, methods, and selection rules of the computer interface. In addition, a fully specified GOMS model provides an opportunity for usability problems to be identified analytically so that aiding dialogues can be implemented effectively based on quantitative predictions of performance time, learning time, and user memory load. Finally, this theory allows assistance and instructional dialogues to be simulated to predict any improvements due to online aiding without extensive user testing.

SECURITY CLASSIFICATION OF THIS PAGE

REPORT DOCUMENTATION PAGE

1a. REPORT SECURITY CLASSIFICATION Unclassified	1b RESTRICTIVE MARKINGS
2a. SECURITY CLASSIFICATION AUTHORITY	3 DISTRIBUTION / AVAILABILITY OF REPORT
2b. DECLASSIFICATION / DOWNGRADING SCHEDULE	
4. PERFORMING ORGANIZATION REPORT NUMBER(S) Tech. Rep. C4E-ONR-1	5. MONITORING ORGANIZATION REPORT NUMBER(S) Tech. Rep. C4E-ONR-1
6a. NAME OF PERFORMING ORGANIZATION University of Michigan 6b. OFFICE SYMBOL (If applicable)	7a. NAME OF MONITORING ORGANIZATION Office of Naval Research
6c. ADDRESS (City, State, and ZIP Code) Center for Ergonomics 1205 - IOE Bldg. Ann Arbor, MI 48109-2117	7b. ADDRESS (City, State, and ZIP Code) 800 N. Quincy Street Arlington, VA 22217-5000
8a. NAME OF FUNDING / SPONSORING ORGANIZATION Office of Naval Research 8b OFFICE SYMBOL (If applicable) Code 1142PS	9 PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER Contract: N00014-87-K-0740
8c. ADDRESS (City, State, and ZIP Code) 800 N. Quincy Street Arlington, VA 22217-5000	10 SOURCE OF FUNDING NUMBERS PROGRAM ELEMENT NO: 61153N 42 / PROJECT NO: RR 04209 / TASK NO: RR0420901 / WORK UNIT ACCESSION NO: 4429008

11 TITLE (Include Security Classification)
(U) Online aiding for human-computer interfaces

12 PERSONAL AUTHOR(S)
Jay Elkerton

13a. TYPE OF REPORT	13b. TIME COVERED	14. DATE OF REPORT (Year, Month, Day)	15. PAGE COUNT
Technical	FROM 87-08-15 TO 88-08-14	87-10-09	40

16 SUPPLEMENTARY NOTATION

17 COSATI CODES		
FIELD	GROUP	SUB-GROUP

18. SUBJECT TERMS (Continue on reverse if necessary and identify by block number)
human-computer interaction, online aiding, assistance dialogues, instructional dialogues, online training, help systems, tutorials, task analysis, cognitive models, GOMS

19. ABSTRACT (Continue on reverse if necessary and identify by block number)
(U) Current research is surveyed on interfaces which aid the computer user online. Results show that many online aiding dialogues fall far short of the ultimate goal of helping users with their current problems, while also supporting continued skill acquisition at the computer interface. To address this problem, a task-analytic approach is presented which is based on the GOMS model (Card, Moran, & Newell, 1983) of human-computer interaction. This model will provide an opportunity for usability problems to be identified analytically, as well as to allow assistance and instructional dialogues to be simulated in order to predict improvements due to online aiding.

20 DISTRIBUTION / AVAILABILITY OF ABSTRACT ☒ UNCLASSIFIED/UNLIMITED ☐ SAME AS RPT ☐ DTIC USERS	21 ABSTRACT SECURITY CLASSIFICATION Unclassified
22a NAME OF RESPONSIBLE INDIVIDUAL John J. O'Hare	22b TELEPHONE (Include Area Code) (202) 696-4502 22c. OFFICE SYMBOL Code 1142PS

DD FORM 1473, 84 MAR 83 APR edition may be used until exhausted. All other editions are obsolete.

SECURITY CLASSIFICATION OF THIS PAGE

Table of Contents

Tables

Figures

CURRENT PROBLEMS IN ONLINE AIDING

This report centers on the design of interfaces that aid the user in computer-based tasks. The goal of these aiding dialogues is to provide details on the procedures for operating the interface which may be unknown to the user. The claim made in this report is that current dialogues for aiding the user of a computer interface are lacking due to fundamental problems with the theories and methods used in the design of these human-computer interfaces.

Ultimately, the objective of an aiding dialogue is to improve both current and long-term user performance with the computer interface. This is an ambitious goal which has not been realized in any aiding application to date. For the most part, aiding interfaces either attempt to support current user performance (i.e., online assistance) or to facilitate long-term user performance and understanding (i.e., online instruction) of a computer-based task (Elkerton & Williges, in press). Although challenging, this goal poses an interesting theoretical question which centers on the possible tradeoff between skill acquisition and performance. More practically, this goal is important since computer users are faced repeatedly with the dilemma of learning additional interface procedures while still trying to complete their current task efficiently.

A variety of aiding dialogues have been developed and tested in the form of online assistance (e.g., help interfaces) and online instruction (e.g., tutorials). Yet, despite this research, little is known about how an online aiding dialogue can be constructed systematically to improve user performance. In fact, it is not uncommon for online aids to increase the time a user requires to solve a problem (Czaja, Hammond, Blascovich, & Swede, 1986; Dunsmore, 1980; Elkerton & Williges, 1984b; Relles, 1979). This report will illustrate these research limitations and will introduce a theoretical approach to construct aiding dialogues which can assist users with their current computer-based task while also fostering continued skill development with the computer interface.

Before reviewing the research literature in detail, some of the global reasons why successful aiding dialogues may be difficult to implement should be stated. One of the most important reasons is that online assistance or instruction is not focused on the user's tasks and goals. Online help, for instance, often is nothing more than an electronic version of a hard-copy manual referencing only the commands and functions of the computer interface. Although potentially useful for skilled users, these online manuals may be completely ineffective for other less skilled or infrequent users who need detailed information on how to operate the computer interface in a specific task. Indeed, Carroll and Mack (1984) have shown that users are active learners who "learn by doing" a computer-based task rather than passively reading documentation or following training materials. Given this task-oriented nature of users, it is not surprising that representative aiding interfaces such as online tutorials may be difficult to use (Carroll & Mazur, 1986).

The second reason for difficulties in developing aiding interfaces may stem from current practices in software interface design. State-of-the-art design processes for aiding interfaces (Al-Awar, Chapanis, & Ford, 1981), and user interfaces in general (Williges, Williges, & Elkerton, 1987), use iterative design techniques with extensive user testing. These design techniques are time consuming and costly since empirical data must be collected from interface users. Consequently, iterative design procedures for developing aiding interfaces may be limited since usability problems that are to be solved with online aiding cannot be identified systematically until after the user interface is fully implemented. Thus, even with iterative development and user testing, the success of the aiding interface is not assured (Carroll & Mazur, 1986).

Finally, one of the basic reasons for poor aiding interfaces is that online aiding is not viewed as an integral part of interface design (Kearsley, 1985). Rather, online user assistance and

instruction is thought of as a remedy for poor interface design. This is probably the result of the philosophical underpinnings of human factors which places emphasis on fitting the interface to the user and the related lack of systematic procedures for designing interfaces to aid the user. Nevertheless, computer interfaces to complex systems demand both an elegant user interface and aiding dialogues to support an unskilled or infrequent user on complicated and demanding tasks. The practical need for online aiding is so strong that at least one human factors text (Bailey, 1982) promotes facilitator design in systems development as a technique to design support materials for the user interface. Although a useful reference, Bailey (1982) does not provide much detail on the procedures for providing support materials online. This report will demonstrate that further research is needed for theories and methods to include online aiding in the interface design process and to complement current procedures for offline user support (e.g., documentation and training).

A SUMMARY OF PROTOTYPICAL ONLINE AIDING DIALOGUES

The scientific literature surrounding online aiding in human-computer interfaces is broad and continues to expand to include applications such as decision aids, expert systems, natural language interfaces, and intelligent tutoring. Despite this research, fundamental questions on the design of these aiding interfaces for human use remain unanswered. To illustrate these limitations, this summary will review the research on online help and computer-supported interface training since these prototypical interfaces demonstrate two general approaches for aiding computer users: aiding through online assistance and aiding through online instruction (Elkerton & Williges, in press). This summary will attempt to illustrate why a theoretical approach to online aiding is necessary. More comprehensive treatments of the online aiding literature are available in Carroll and McKendree (1987) and Elkerton and Williges (in press).

The goal of an assistance dialogue is to reduce the user's effort in accomplishing the current computer-based task. Typically, online documentation, procedures, prompts, and cues are provided to support user performance. Help systems, for example, provide the user with additional details on possible functions and commands of the computer interface. Theoretically, this supporting information will reduce the amount of learning, memory, and cognitive processing required for the computer-based task resulting in a decrease in user time and errors, and the need for user training and selection (Bailey, 1982).

Alternatively, instructional dialogues train and educate the user to increase knowledge of the computer-based task and facilitate long-term performance. Thus, an instructional dialogue may sacrifice current performance to improve task comprehension. These dialogues often encourage the user to engage in additional exercises and practice to develop a deeper understanding of the task. Examples of these instructional dialogues include computer-supported interface training (Al-Awar, et al., 1981; Carroll & Mazur, 1986), computer-based instruction (Eberts & Brock, 1984; Robinson & Knirk, 1984), and intelligent tutoring (Sleeman & Brown, 1982).

Online Assistance Dialogues

The most widely available assistance interface is the online help system. As discussed by Shneiderman (1986), providing online help is attractive for several reasons. First, the user does not need hard-copy manuals which take up workspace and possibly divert the user's attention from the video display and computer-based task. Second, online help documents can be modified relatively easily. Third, online help if designed with electronic indexes and cross-references, can enhance the user's capability to retrieve documentation quickly. Finally, new graphics technology in the form of diagrams and animation may aid users in understanding the interface structure which may help users learn and remember computer-based procedures.

As a result of these possible benefits, there are a variety of help dialogues that have been developed and identified by Borenstein (1985), Elkerton and Williges (in press), and Houghton (1984). Table 1 presents a brief list of these help dialogues with appropriate references. Clearly, there is no technology shortage for presenting help information. Unfortunately, few behavioral investigations have been conducted on these help interfaces. And more importantly, few studies have considered seriously how these help interfaces fit into the user's computer-based task.

Online manuals are good examples of problems with help interfaces. As mentioned previously, help interfaces are often nothing more than online versions of hard-copy manuals with keyword or menu-based mechanisms for accessing the help documentation. However, this alone may not be problematic. The problem with electronic help manuals is that the knowledge represented and accessed through the help interface is inadequate for many computer-based tasks a user faces. Most users do not want a hierarchical list showing the syntax of a command. Few users need this detailed, fact-oriented knowledge. Instead, many computer users need to know the methods to complete a task. Without this procedural knowledge (knowledge of how to do things), users are left to browse through a wealth of information with little understanding of what help topics may be useful. Indeed, some of the initial studies which found performance decrements with online help (Dunsmore, 1980; Relles, 1979) may have been related to the poor procedural content of the help dialogues and the resultant searching behavior which disrupts the computer-based task.

Similar problems with an online manual for novice users were demonstrated by Cohill and Williges (1985). These investigators evaluated eight types of help determined by help format (online or hard-copy), help initiation (user or computer), and selection of help topic (user or computer). The study compared these help conditions to a control condition where novices received no help in the text-editing task. The results showed that all help conditions were

superior to the no help control group. However, the help systems which yielded the best performance in terms of time and errors were conditions where novices initiated and selected help material from a hard-copy manual.

Table 1

Dialogues for Online Help

1.	Online manuals (Cohill & Williges, 1985; Dunsmore, 1980; Relles, 1979)
2.	Minimal help texts (Carroll, Smith-Kerker, Ford, & Mazur, 1986)
3.	Keyword help (Borenstein, 1985; Houghton, 1984; Magers, 1983)
4.	Menu-based help (Borenstein, 1985)
5.	Query-in-depth (Houghton, 1984)
6.	Concrete and simulated examples (Grignetti, Hausmann, & Gould, 1975; Magers, 1983)
7.	Command prompting (Mason, 1986)
8.	Context-sensitive help (Borenstein, 1985; Fenchel & Estrin, 1982; Magers, 1983)
9.	Task-oriented help (Carroll, et al., 1986; Finin, 1982; Magers, 1983)
10.	Window-based help (Borenstein, 1985; Orwick, Jaynes, Barstow, & Bohn, 1986; Teitelman, 1985; Walker, 1985)
11.	Diagrammatic help (Sebrechts, Deck, & Black, 1983; Shneiderman, 1986)
12.	Natural language help (Borenstein, 1985; Wilensky, Arens, & Chen, 1984)
13.	Intelligent online help (Aaronson & Carroll, 1987; Fischer, Lemke, & Schwab, 1985)

One of the conclusions drawn by Cohill and Williges (1985) was that hard-copy manuals allowed users to browse through help information while keeping the text-editing task on the display. Still, users should not have to browse help information since this is time that users could spend learning the computer-based task. If the online help information is condensed into what the user needs to operate the interface, then less time will be spent looking for

information. Less display space also will be required so that both the computer-based task and the online help can be presented simultaneously. Unfortunately, this online help research offers little in terms of how help content could be designed to focus on the methods for operating the user interface.

In current online help research what is typically presented are further technical solutions to these display problems. For example, large-screen, bit-mapped displays now make it possible to display large amounts of help information with the computer-based task. Examples of this window-based help can be found in systems described by Orwick, et al. (1986), Teitelman (1985), and Walker (1985). These window-based help systems may relieve the short-term memory of the user by allowing visual comparison of the help file with the computer-based task. Thus, the user's cognitive processing has been considered with window-based help, but not rigorously. For example, consider the situation where five windows are open on a screen. In this situation, the help window probably will have to overlap other windows or the sizes of other windows will have to be reduced. Consequently, the user may still not be able to see all the necessary information and may have to execute additional procedures to manipulate these windows on the display. As Shneiderman (1986) has proposed, introducing a help window actually may increase the cognitive load on the user.

Clearly, the methods for interacting and accessing help information is a significant problem with user assistance and online help (Sondheimer & Relles, 1982). As an example, Borenstein (1985) found that providing several access methods (i.e., window-based help, keyword help, computer-initiated help, menu-based help, and context-sensitive help) in a single interface called ACRONYM improved the performance of expert and novice users when compared to natural language help and a standard online manual. In fact, the only help method shown to be superior to ACRONYM was a human tutor. Borenstein (1985), however, was unable to isolate whether the help access methods or the quality of the help text improved user performance.

On this very point, research conducted by Carroll, et al. (1986) has emphasized strongly the need for well designed help texts in the form of minimal manuals. These minimal manuals attempt to reduce the verbiage of help texts, support error recognition and recovery, and focus documentation on real tasks and activities. Applying these general principles for designing help texts has demonstrated significant improvements over standard help manuals (Black, Carroll, & McGuigan, 1987; Carroll et. al.). However, the specific procedures for adhering to these guidelines for minimal manuals are not well developed and additional research is needed to develop theory-based methods for constructing the substantive content of online help texts.

As a further example of current research limitations with online help, consider the study conducted by Magers (1983) on the effectiveness of a variety of help methods. A current help interface in this study was enhanced by: (1) introducing a help key, (2) providing more examples and a more concrete help text, (3) making the help more task-oriented, (4) providing command synonyms and context sensitive help, (5) suggesting specific correction methods in error messages, (6) allowing several methods for formulating help requests, (7) minimizing the length of help, and (8) introducing feedback to confirm legal commands. User performance and attitudes toward this enhanced help interface improved significantly. Thus, help interfaces can be improved. However, this study failed to determine which help method had the largest impact on performance.

Beyond this literature, very little additional research exists on the behavioral efficacy of online help systems. Dominating the literature are technical approaches to provide sophisticated help interfaces to users. Examples of these efforts are the context-sensitive help interfaces developed by Fenchel and Estrin (1982), adaptive command-prompting interfaces (Mason, 1986), help systems that provide concrete simulations relevant to the user's task (Grignetti, et al., 1975), help interfaces with natural language front-ends (Wilensky, et al.,

1984), and knowledge-based approaches to provide more intelligent online help (Fischer, et al., 1985). These technical efforts explore the feasibility of new help interfaces. However, few if any of these efforts focus on the user's needs for online assistance.

Addressing these user needs, Aaronson and Carroll (1987) have observed and analyzed the dialogues between a user and human consultant to determine what inquiries and responses will be required with more intelligent online help. In studying the verbal protocols of one-time consultations through computer mail, these investigators identified several strategies which may be useful in future help systems. These strategies included: making explicit assumptions about user goals, providing alternative solutions, assuming an interface configuration, avoiding the problem, and pointing to reference sources. Although a useful and creative technique for determining requirements for intelligent online help, the analysis of verbal protocols (not unlike other data analysis techniques) is time consuming and requires considerable skill on the part of the analyst. In addition, as will be illustrated later in the report, at least two of the consultations would be expected from a theoretical analysis (making assumptions about goals, and providing alternative solutions, i.e., methods for operating the user interface). These types of observational approaches should be augmented with more theoretical methods to focus data analysis and to assure that empirical data are interpreted appropriately.

Table 2 summarizes the research on online help. Based on this review, help interfaces can be designed to improve user performance. However, the help content and dialogues that actually contribute to this improvement are relatively unknown. Thus, it is not surprising that some of the investigations illustrate disruptive effects associated with online help since there is a lack of detailed procedures for developing the help interface with only qualitative principles (e.g., make the help more task oriented) to guide the designer. As result, the development of the help interface must often proceed without any knowledge of the improvements in usability that will result from the aiding dialogue. Without this, designers will be forced to adopt a technology-

driven approach for help interfaces which may or may not fit the needs of the user. What is needed is an understanding of the interface methods that the user needs to extract from online help and how this procedural information can be conveyed by the aiding dialogue without disrupting the computer-based task. Moreover, this method-based information should be in the form that designers can use with a minimum of time-consuming user testing.

Table 2

Summary of the Research on Online Help

---

1. Behavioral investigations are limited considering that online help is a major component of many user interfaces.
2. Online help can be improved, however, a few investigations have illustrated the potential for the disruptive effects of online help.
3. What little research has been done has generated qualitative principles for online help, but has not identified clear procedures for developing and predicting the possible improvements with help interfaces.
4. Currently, the primary focus is on the computing technology for developing new help interfaces.
5. Little attention is paid to the method-based content required by the user of the help interface.
6. Research and design of help interfaces, at this point, is tied to extensive user testing.

Online Instructional Dialogues

In contrast to the limited behavioral research on online help and user assistance dialogues, instructional dialogues have a larger research base ranging from computer-based instruction (CBI) to intelligent tutoring (IT). However, no attempt will be made to summarize the vast literature surrounding CBI or IT. Rather, the emergent area of computer-supported interface training will be addressed since this research focuses on users acquiring interface skills. Table 3 lists some of these dialogues for interface training and appropriate references.

The most common example of computer-supported interface training is the online tutorial. These programs attempt to provide general purpose instruction and practice on interface procedures. To accomplish this, tutorials typically provide some relatively simple tasks that users can practice at the interface. As an example, Al-Awar, et al. (1981) using iterative design methods have developed and evaluated tutorials for acquainting new users with a computer terminal. The online tutorials allowed users to focus directly on the keyboard and display without switching their attention to an instructional manual. In addition, low-level interaction skills which could transfer to other interface tasks (e.g., using specific keys) were emphasized in these tutorials. Thus, tutorials may help in the acquisition of fact-related information and basic procedural skills by focusing on specific interface operations (finding and pressing keys) as opposed to more general interface methods (determining the correct sequence of operations to accomplish a task).

Table 3

Dialogues for Computer-Supported Interface Training

1. Online tutorials (Al-Awar, et al., 1981; Carroll & Mazur, 1986; Czaja, et al., 1986)
2. Guided exploration and problem solving (Carroll, Mack, Lewis, Grischkowski, & Robertson, 1985; Charney & Reder, 1986)
3. Training wheels interfaces (Carroll & Carrithers, 1984; Catrambone & Carroll, 1987)
4. Scenario machines (Carroll & Kay, 1985)
5. Command-selection aids (Elkerton & Williges 1984b; 1987)

In fact, a study by Czaja, et al. (1986) has demonstrated that traditional tutorials can be limited in their capability to train procedural skills beyond simple operations. These

investigators looked at three possible training strategies for a word-processing program (Wordstar) which included online tutorial training, document-based training, and instructor-based training. The study found that users of online tutorials performed much worse on the transfer tasks than users who received document and instructor-based training. The users of online tutorials accomplished fewer transfer tasks, took more time, and committed more errors. Interestingly, the errors committed by these tutorial users on the word-processing tasks indicated that they were not familiar with editing procedures such as inserting, deleting or rearranging text. Tutorials were not effective in helping users learn procedures which are necessary for actual word-processing tasks.

Why might traditional tutorials be limited in teaching general interface methods? It may be that interface methods are not simply sequences of keystrokes, but a set of actions to accomplish a user's task. Since traditional tutorials provide the materials and tasks for the user to practice, they fail to integrate the interface actions with meaningful user goals. Thus, tutorials require the user to practice tasks that are foreign to the actual work to be performed, and as a result, may be passive learning devices which do not allow the user to actively explore interface methods. In fact, Carroll and Mack (1984) have noted that many inexperienced users strike out on their own, learning by doing a computer-based task rather than using a potentially time-consuming tutorial.

Testing these hypotheses, Carroll, et al. (1985) have studied offline tutorials and "guided exploration" manuals that are task-oriented, procedurally incomplete, modular, and capable of supporting error recognition and recovery. Users of the guided exploration manuals spent more time learning than users of the traditional tutorials. However, in transfer tasks users that received the guided exploration manuals completed the tasks more efficiently and correctly than tutorial users. Carroll, et al. argue that the success of this approach is related to reduced set of instructional materials which focused the user on problem solving and exploration with the

word-processing interface. In addition, Carroll and his colleagues state that guided exploration provided users with much more information on how to recover from task-related errors. When compared to a guided exploration users, traditional tutorial users spent more time and effort dealing with errors while learning.

Further support for a problem-solving orientation when training users has been provided by Charney and Reder (1986). These investigators found that users who solved task-oriented problems when learning a spread-sheet interface did much better on new problems than users provided step-by-step tutorial instructions. However, similar to the Carroll, et al. (1985) study, the tradeoff for this improved transfer performance was additional time for users to solve the problems during training. As discussed previously, tutorials are limited in teaching procedural skills since users may have difficulty applying the step-by-step procedures and specific examples to realistic tasks. Yet, the alternative of supporting active learning also has a cost in terms of additional training time.

One solution to this tradeoff may be to exploit active learning in the actual interface. Carroll and Carrithers (1984), for example, have developed a training-wheels interface which allowed new users to explore a word-processing interface while doing meaningful tasks. This modified word processor supported initial user learning by blocking advanced features and allowing only basic word-processing operations (i.e., creating, editing, and printing documents). The results of this investigation found that more users of the training-wheels interface completed a letter printing task and did so faster than users of a complete word processor. Carroll and Carrithers explained the success of the training wheels approach through users gaining feedback without suffering the negative consequences of error states. In addition, Carroll and Carrithers noted that the training-wheels interface may be a more effective learning environment since the number of system functions was limited and allowed users to form and test hypotheses regarding word-processing methods in a smaller domain.

Extending and validating these results, Catrambone and Carroll (1987) have compared the training-wheels interface to a complete word processor both during learning and a transfer task where users attempted to use additional word-processing features. As in the previous investigation, the training wheels interface helped users complete their training faster than users of a complete word processor. More significantly, the training wheels interface assisted computer users in learning additional functions of the word processor faster than users of the complete word processor.

Catrambone and Carroll (1987) ascribe this improvement in training and transfer to the reduced complexity of the training wheels interface which allowed users to more fully understand and apply the basic interface methods to the advanced procedures of the word-processing software. Specifically, many of the common menu skills learned by users in the training wheels system may have transferred to the advanced functions. This result comes closest to the goal of designing an aiding interface which allows users to complete current computer-based tasks while also encouraging further skill development with the interface. The success of this progressive exposure to the user interface demands that theoretical models be developed for identifying the common methods for staged interface training. That is, the designer needs systematic procedures to determine which methods can be used in learning more complex interface skills.

In a similar extension of the training wheels approach, Carroll and Kay (1985) have implemented and evaluated several "scenario machines" where a new user of a word-processing system is guided through a single method or scenario for creating and printing a document. The goal of this research was to determine what combination of user prompting, feedback, and error correction facilitated training and transfer in a word-processing task. Overall, the results from Carroll and Kay were in agreement with the training-wheels research (Carroll & Carrithers, 1984). Scenario machines which guided the user in using a single method for

creating and printing a document reduced training time significantly when compared to a control group with a complete word processor.

However, Carroll and Kay (1985) found that transfer performance for users of scenario machines did not differ from those users with a complete word processor. In fact, prompting and feedback in some scenario machines was damaging to transfer performance. This suggests that the training dialogue (i.e., additional prompts and feedback for user actions) could have destroyed the coherence of the computing tasks. The training dialogue, whether it be staged disclosure of the interface methods or explicit training prompts and cues, must be carefully specified and evaluated to assure an effective aiding interface. Indeed, a challenge for computer-supported interface training is to develop theoretical methods which will predict the usability of these online aiding dialogues. With theoretical models, aiding dialogues which might interfere with user performance could be identified quickly and other more promising aiding dialogues targeted for further refinement.

As a final example of computer-supported interface training, Elkerton and Williges (1984b; 1987) have investigated a variety of command-selection dialogues for improving the strategies and performance of novice users in file-search environments. With these aiding dialogues, novices were presented command-selection advice that prompted novices to use more powerful search commands typically selected by experienced users.

In the first experiment, Elkerton and Williges (1984b) demonstrated that novice computer users could be trained to learn more sophisticated search commands when automatically provided hints on the frequency of command use by more experienced users. However, the price paid by users with this computer-initiated, command-selection advice was an increase in time and effort to interpret and use this advice. Admittedly, advising novices by presenting the frequently selected commands of more experienced users is an impoverished

aiding dialogue that contains no procedural, command-sequence, or planning information for file search (Elkerton & Williges, 1985). However, the research demonstrates that novices could use this limited command-selection feedback to facilitate skill development with search strategies.

In an extension of this research, Elkerton and Williges (1987) developed additional command-sequence and plan-based models and explored alternative aiding dialogues (i.e., user-initiated, computer-initiated, and mixed-initiated - both user and computer initiated) in an attempt to alleviate some of the intrusiveness of the command-selection advice. In other words, the research goal was to diminish the apparent tradeoff between novice acquisition of more powerful search skills and their current file-search performance. Similar to the results of Elkerton and Williges (1984b), command-selection aids improved the search performance and strategies of slow novices when the command-selection models were not unduly complicated (i.e., frequency models and plan-based models) and when the online aiding did not introduce a highly interactive, mixed-initiative dialogue.

These command-selection experiments (Elkerton & Williges, 1984b; 1987) illustrate the variety of methods for instructing users on interface procedures ranging from simple hints on frequently used commands to step-by-step plans of command-selection. Unfortunately, the procedures for constructing these command-selection models are based on collecting and analyzing the command-selection performance of expert users. Such an empirical strategy requires substantial time and effort to build the command-selection models and is not practical for a new user interface. Thus, a systematic procedure to specify the command-selection model would be useful. In addition, similar to the implications drawn from the research on scenario machines (Carroll & Kay, 1985), the intrusiveness of the mixed-initiative dialogue underscores the need for theoretical models to predict when online aiding will not lead to improved performance at the user interface.

Table 4 summarizes the research on computer-supported interface training. As illustrated by this brief review, there is no lack of data. Moreover, the results from the literature are encouraging and suggest some basic principles for assisting users both with short-term learning and long-term skill acquisition. For example, principles such as reduce the complexity of the initial training interface, decrease the possibility for user errors, and progressively expose the user to the interface methods are sound recommendations for developing aiding dialogues based on well-known learning principles. In fact, these recommendations are based on age-old learning principles: providing positive reinforcement (achieving task goals) rather than punishment (making errors at the user interface) to improve learning (Thorndike's truncated law of effect) and using part-whole training paradigms. Not surprisingly, there are many other learning principles (see Kyllonen & Alluisi, 1987) which are applicable and should be used to lessen the required testing of users during design of computer-supported interface training. However, the principles need to be placed within a usable framework so that the method-based content of instructional dialogues can be developed along with theoretical usability predictions for the aiding dialogue.

Table 4

Summary of the Research on Computer-Supported Interface Training

---

1. Behavioral investigations are available that demonstrate that computer-supported interface training can decrease the training time and also promote skill transfer.

2. These behavioral investigations have generated a set of principles based on theories of human learning.

3. These principles, however, do not provide the designer with the appropriate methods to determine what should be trained and do not provide a mechanism for predicting the usability of an instructional dialogue.

4. Research on interface training is empirical in nature requiring the instructional interfaces to be tested extensively with users.

A RESEARCH AND DESIGN FRAMEWORK FOR ONLINE AIDING

Given the limitations to the online aiding approaches a more formal and theoretical model should be adopted for the detailed development of these dialogues. As outlined in Table 5, this theoretical model should: (1) describe how the aiding dialogues should be implemented, (2) determine when online aiding is necessary in a human-computer interface, and (3) be capable of predicting the effectiveness of the aiding dialogues without a complete implementation of the user interface. With this theory-based approach, development of aiding interfaces should proceed quickly from initial design to final implementation due to more formal analysis and less trial and error testing of the aiding dialogues. In addition, developing this theory of online aiding should provide a more systematic method for evaluating aiding interfaces since it will be possible to identify and separate the procedural knowledge presented by online aiding from the dialogue used to deliver this assistance or instruction.

Table 5

Requirements for a Theory of Online Aiding

• The theory should be capable of providing the substantive content for the aiding dialogues.
• The theory should be capable of predicting when and where online aiding is necessary through analysis of user interface tasks.
• The theory should be capable of predicting any improvements in usability which will result from the aiding dialogues.

A Theory-Based Task-Analytic Model for Online Aiding

Describing how aiding dialogues should be implemented requires a task analysis. Certainly, recommending that a task analysis be conducted before developing training materials (Goldstein, 1987) or job aids (Swezey, 1987) is not a new idea and is a major part of Instructional

Systems Development (see Meister, 1985). However, using a cognitive task analysis which focuses on the procedural knowledge required to operate a computer interface in detailed development of online aiding is a relatively new and powerful concept. In fact, a great deal of research has been conducted in cognitive psychology focusing on this knowledge and has resulted in a computer modeling approach in the form of production systems (see Anderson, 1983). These rule-based systems have the capability to represent symbolically human procedural knowledge to simulate and predict human performance. Performing this type of cognitive task analysis will allow the identification of detailed interface procedures which could be used as the substantive content for the aiding dialogues and will also introduce a capability to predict human performance with these aiding interfaces.

The task-analytic approach for defining the cognitive procedures that a user must perform at the computer interface is best summarized by Card, Moran, and Newell (1983). These investigators developed a model for describing the user's interface knowledge in terms of **Goals** (what the user must accomplish), **Operators** (the individual actions, such as pressing a key or moving a mouse), **Methods** (step-by-step procedures for accomplishing goals), and **Selection Rules** (heuristics for specifying which method to use in specific circumstances). This **GOMS** model is gaining acceptance as a method for analyzing what the user must know to operate the computer interface. Moreover, the importance and power of this approach for online aiding is clear since GOMS is a top-down specification of the procedural knowledge that could be used to specify online aiding during interface design. That is, not only can GOMS models be used to design a user interface, but the goals, operators, methods, and selection rules also can be used to provide online assistance and instruction, if necessary. In the following sections examples of online aiding for each component of the GOMS model will be presented.

Using goals in online aiding. Identifying the goal structure of a task is a commonly advised strategy for tutoring students in new problem-solving domains (Anderson, Boyle,

Farrell, & Reiser, 1984) and for assisting computer users during sessions with a human consultant (Aaronson & Carroll, 1987). Consequently, goal-level aiding is particularly appropriate for advising new users on what can be accomplished at a computer interface. For example, as suggested by Elkerton (1987) the help system of word-processor could provide a list of the goals so that a new user could quickly learn what can be accomplished with the interface (see Figure 1). Indeed, providing this goal-level advice was one of the guiding principles in the successful design of training wheels interfaces (Carroll & Carrithers, 1984; Catrambone & Carroll, 1987) and minimal manuals (Carroll, et al., 1986).

With this word-processor you can:

Type a manuscript
Edit a manuscript
Print a manuscript
Delete a manuscript

Type the first letter to get more information on these tasks.

Figure 1. *Goal list provided in a help interface for a hypothetical word-processor (adapted from Elkerton, 1987).*

Goals as online advice describe to the user what can be done at the interface in a task-oriented language (Carroll, et al., 1985). Ideally, goal-level aiding should enable users to transfer appropriate procedural knowledge from the task domain to learning how to use the interface. Users with this type of online advice should understand quickly the procedural objects and actions of the interface. According to Charney and Reder (1986), this knowledge of basic concepts and functionality of the interface is a critical component for initial skill learning.

As shown in Figure 1, an important feature of a goal tree for online aiding is that the hierarchical structure can be used to describe additional procedural details by accessing subgoals for each high-level goal. For example, subgoals such as find a string and move a line could be presented below the higher-level goal of edit a manuscript. Thus, the interface structure is easily conveyed to the user in terms of what the user can accomplish. Catrambone and Carroll (1987), for instance, state that one of the advantages of the training-wheels interface was that users were made aware of the procedural structure of the interface.

This multi-level description is also convenient for providing users an adjustable level of detail on interface procedures as recommended by Anderson, et al. (1984). Thus, experienced users only may need to be reminded that a word-processor allows you to move text, while a less experienced user may need additional information on the subgoals of moving text such as select, cut, and paste text. In addition, Carroll, et al. (1985) suggest that providing information on the goals and subgoals would be procedurally incomplete advice to be used actively by users. With this information users could explore (Carroll, et al.), elaborate (Reder, Charney, & Morgan, 1986), problem solve (Charney & Reder, 1986), or make inferences about interface methods (Black, et al., 1987) to improve their understanding and performance in the computer-based task.

Goals as online advice also provides a convenient mechanism for presenting feedback to the user on their interface actions and may be useful for error recovery. The training wheels interface of Carroll and Carrithers (1984) demonstrates this point. Since only a limited set of interface goals was available, users were provided immediate feedback on the correctness of their actions. Moreover, if users committed an error, they were able to recover from the error relatively easily and subsequently were reminded through the menu which word-processing goals were appropriate. That is, goals as online advice can serve to keep users focused on the task so that they do not become entangled with other interface details. This has been confirmed

by McKendree and Carroll (1987) who found that error-blocking with goal feedback helped users complete computer-based office tasks faster and also increased the number of users who completed a transfer task.

Finally, using high-level goals as advice may provide modularity so that assistance and instruction can be focused on relatively independent interface tasks (Carroll, et al., 1985). Using goals provides a focus for the aiding dialogue so that assistance and instruction on specific tasks can be separated from other interface knowledge. In other words, formalizing the procedural knowledge in a goal tree can form the basis for aiding techniques such as learning by parts and by successive approximations (Anderson, et al., 1984). Thus, the decomposition of the procedural knowledge defines what should be presented in online aiding and also provides an effective diagnostic tool for determining what the user is trying to accomplish at the interface.

Using operators in online aiding. Presenting operators as online advice is more or less standard practice with online help and tutorials. Specific examples of operator-level aiding include help systems which provide command lists (see Figure 2) and tutorials which ask users to perform simple actions such as finding and pressing keys. With this impoverished type of online aiding, many online help systems (Dunsmore, 1980; Relles, 1979) and online tutorials (Carroll & Mazur, 1986; Czaja, et al., 1986) are difficult to use since operators are only a small part of the procedural knowledge required to operate the interface. Moreover, as illustrated by Figure 2 presenting all of this operator knowledge can be overwhelming to a novice user making it a difficult task to select the appropriate topic for online assistance. Application of the GOMS model would suggest that additional knowledge must be provided such as which object the operator acts on (e.g., open a file), what other actions follow or precede the present operator, and where a specific operator is used in a computer-based task. In short, operator-level aiding probably should be augmented with knowledge about goals, methods, and selection rules.

Nevertheless, operator-level aiding may be useful if the costs for current user performance are understood. Specifically, operator-level aiding frequently requires additional time and effort for the user to process and apply this knowledge in their computer-based task. Users actively processing this information can experience improvements in long-term performance and understanding. An example of this tradeoff of current performance with future task performance can be seen in the research conducted by Elkerton and Williges (1984b; 1987) on command-selection aids for novice users. In these studies, dramatic increases in task times frequently were observed when providing command-selection advice. However, after receiving this command-selection advice novices enriched their command-selection strategies, and consequently improved their long-term performance.

MTS_Commands_Menu

Type the number or name of one of the MTS commands listed in the menu below to get information about that command.

1	ACCOUNTING	2	ALTER	3	CALC	4	CANCEL	5	COMMENT
6	CONTROL	7	COPY	8	CREATE	9	DEBUG	10	DESTROY
11	DISPLAY	12	DUMP	13	DUPLICATE	14	EDIT	15	EMPTY
16	EXPLAIN	17	FILEMENU	18	FILESTATUS	19	HELP	20	IF
21	LIST	22	LOAD	23	LOCK	24	LOCKSTATUS	25	LOG
26	MAKE	27	MESSAGESYSTEM	28	MODIFY	29	MOUNT	30	$MTS
31	NET	32	PERMIT	33	RELEASE	34	RENAME	35	RENUMBER
36	RERUN	37	RESTART	38	RUN	39	SDS	40	SET
41	SIGNOFF	42	SIGNON	43	SINK	44	SOURCE	45	START
46	SYSTEMSTATUS	47	TRUNCATE	48	UNLOCK	48	UNLOAD		

Press CRTL-E to leave HELP mode. CRTL-C for help on using the screen.
NumericPad 1/END to enter a topic name or number you have typed or the key alone for the previous topic (MTS_COMMANDS)
Topic selection:

Figure 2. *Example of operator-level help in the form of command lists in the Michigan Terminal System - MTS (University of Michigan Computing Center, 1984).*

In addition, operator-level aiding may be highly appropriate for more skilled users. In these aiding applications, users may have the requisite procedural knowledge to understand

suggestions for detailed interface operations. An example of this operator-level aiding for more skilled users is the "Did You Know" interface implemented by Owen (1986) to provide users with information on novel command applications which were suggested by other users. This operator-level aiding for more experienced users, however, needs further research to determine its potential usefulness.

A final use of interface operators in online aiding concerns the monitoring of user actions. Operators such as keystrokes and mouse movements are typically the only behaviors which can be monitored directly at the user interface. As an example, the last keystroke or command must be monitored with context-sensitive help (Borenstein, 1985; Fenchel & Estrin, 1982). However, it must be emphasized that monitoring only at the operator level may not be the most effective strategy for online aiding. Research support for this can be seen in the work of Elkerton and Williges (1984b) who found that online advice could be presented at very inappropriate times when monitoring novices on infrequently used commands. In addition, extending this operator monitoring strategy to second-order command transitions was equally or even more disrupting to novices (Elkerton & Williges, 1987). Thus, monitoring operators will probably require accessing additional procedural knowledge in the form of goals, methods, and selection rules.

Using methods in online aiding. Providing methods as online advice is one of the more obvious techniques for aiding and was suggested by Aaronson and Carroll (1987) in the analysis of dialogues between computer users and human consultants (i.e., providing alternative solutions). The value of this online aiding is that the step-by-step procedures can be presented to the user to solve specific interface problems. Until recently method-level advice has not been incorporated into many aiding interfaces. However, new user interfaces such as the Digital VAXstation II (Digital, 1986) and the Xerox ViewPoint workstation (Xerox, 1986) are

beginning to include method-level help for such activities as window management. An example of this method-level help has been presented by Elkerton (1987) and is shown in Figure 3.

To move a window:

1. Place the arrow inside the window you want to move.
2. Hold the LEFT mouse button and move the mouse to re-position the window.
3. Release the mouse button.

Figure 3. *Help method for moving a window in a hypothetical computer workstation (adapted from Elkerton, 1987).*

Based on the reviewed research, methods appear to be well suited for helping the user with their current computer-based task (online assistance), but may be limited in helping users acquire skills for improved long-term performance at the user interface (online instruction). For example, both Carroll and Kay (1985) and Charney and Reder (1986) have found that training time is substantially reduced when users are provided a single step-by-step method. However, in both of these investigations advanced performance in a transfer task was not improved significantly as a result of a receiving a single interface method. In addition, Elkerton and Williges (1987) have found that novices prefer step-by-step procedures while learning a file-search task, but did not improve their transfer performance beyond other novices receiving less complete command-selection advice (frequency and sequence of file-search commands).

The capability for method-level advice to improve current interface performance is an important characteristic and should be explored further. For example, McKendree and Carroll (1987) found step-level feedback (what users should do next) after blocking errors with computer-based office tasks similar in effectiveness to goal-level feedback (what users are

trying to accomplish). As pointed out by Elkerton and Williges (1987), method-level advice may be useful in reducing the users' cognitive load while learning a new task. However, this benefit must be weighed against the problem of "spoon-feeding" users rote procedures which may not lead to additional acquisition of interface skills (McKendree & Carroll).

The problem with method-level knowledge for skill acquisition may be that a method is too specific for users to apply beyond the current problem-solving context. Further research must be conducted to determine how methods could be generalized so that learners can use this procedural knowledge to solve their current problem, while also acquiring skills which will help them in future tasks. In fact, current research by Lewis, Casner, Schoenberg, and Blake (1987) has introduced the concept of learning from a procedural demonstration. The thesis of this research is that users are capable of learning interface methods from simple demonstrations of command use and often do not need explicit procedures or multiple examples for acquiring interface skills. The heuristics for identifying interface methods that can be effectively demonstrated are just being developed (i.e., mapping rules between user actions and system responses). However, the use of procedural demonstrations may be a powerful learning technique if principles can be validated for choosing the methods to be demonstrated.

Similar to goal-level advice, the grain-size of the method presented to the user is critical to the ultimate success of online aiding (Anderson, et al., 1984) and should be investigated within the hierarchical GOMS model. In addition, alternative mechanisms for conveying this method-based knowledge to users should be investigated. In particular, graphical flow diagrams (Bauer & Eddy, 1986; Sebrechts, et al., 1983) seem appropriate since method-based knowledge consists of a series of procedural steps and decisions. With a graphical presentation, users may be better able to remember method-based knowledge.

Interface methods also may be useful from the monitoring perspective in online aiding. Ideally, a user's methods could be tracked and alternate methods suggested when appropriate. An example of this method-based monitoring has been developed by Finin (1982) as an online help system called WIZARD. In brief, WIZARD attempted to diagnose inefficient user plans for executing commands in an operating system. For instance, if the user was deleting old versions of a file repeatedly with "delete", WIZARD would detect this activity and suggest an alternative and more efficient interface method such as purging old files in a single step. Obviously, monitoring and inferring appropriate interface methods is a complicated activity which requires further research. In fact, novel solutions such as asking the user what they are attempting to do from a reduced set of inferred methods may reduce some of the reasoning that may be needed in an online aiding system.

Using selection rules in online aiding. Unlike the previous components of the GOMS model, selection rules are seen rarely in online aiding, but may be necessary when there are multiple interface methods and when the user is unaware of the reasons for using a specific method. As an example of a situation in which selection rules might be helpful, consider the word-processing task where users have to search a file for text to be modified. In these search tasks, Elkerton and Williges (1984a) have found large differences in strategies of novice and experienced users indicating that these two groups of users may have different rules for choosing commands. Novices scrolled and paged through a file, while more experienced users also selected string search procedures. Consequently, novices may have been unaware of the appropriate selection rule for using the string search procedures. A selection rule which Card, et al. (1983) have formulated for file search states that scrolling methods should be used when the estimated distance to the text is relatively small (e.g., less than 4 lines) and string search methods should be used when the distance to the text is relatively large (e.g., 4 lines or greater). Novices may benefit by having a selection rule like this presented to them. Indeed,

Charney and Reder (1986) have stated that an important component in skill acquisition is learning the context in which a procedure is applied.

Interestingly, an alternative approach to providing the selection rules as advice is to present appropriate problem-solving examples for user skill acquisition. Elkerton and Williges (1984b; 1987), for example, aided novices in the acquisition of command-selection strategies without explicitly providing selection rules, but by prompting novices to use an appropriate command in a representative search task. Novices improved their search strategies with this advice. However, as mentioned previously, the cost of this approach was a decrease in performance during the online aiding. Reder and Charney (1986) argue similarly that the success of their problem-solving approach when training novices on spread-sheet commands may have been due to users learning when to apply the commands.

Therefore, the question which remains based on this theoretical treatment is whether explicitly presented selection rules are superior to presenting appropriate task examples so that the user can infer and develop their own selection rules. The challenge, perhaps, is to develop an effective means for communicating the relevant features of selection rules. Possible presentation techniques include the previously mentioned flow diagrams where selection rules could be presented in a diagram illustrating the interface method graphically. Similarly, other display techniques such as decision tables (Gettys, 1986) should be considered for these "if-then" selection rules.

Summary: Using GOMS models in online aiding. The thesis of this section has been that components of the GOMS model could serve as the substantive content of aiding dialogues, while also providing a theoretical basis for evaluating these dialogues. To summarize, Table 6 presents a set of suggested principles for the design of online aiding interfaces based on what is known about each component of the GOMS model. For the most part, these principles are only

working hypotheses that require further experimental validation. Consequently, the designer must interpret these suggestions with care, while also monitoring the new empirical research in this rapidly evolving area.

As illustrated in Table 6, there are numerous design suggestions which emerge by looking at each individual component of the GOMS model. However, online aiding would probably be best served if these model components and dialogues were combined into a single aiding interface. Goals, operators, methods, and selection rules all should be incorporated into an aiding interface either in presenting this procedural knowledge to users or in using this knowledge to monitor and diagnose problems that a user may have in operating the interface. Taken together, these GOMS models may be similar to the plan-based models proposed by Riley and O'Malley (1984) and Hayes and Williges (1986) for understanding human-computer interaction.

Predicting Usability for Online Aiding

The GOMS task-analytic approach for specifying the procedural details of aiding dialogues also can provide two necessary ingredients for effective online aiding interfaces shown in Table 5. First, given a GOMS specification, detailed quantitative models, such as Keystroke models (Card, et al., 1983) and production rule systems (Kieras & Polson, 1985) can be used to predict usability problems with the interface which could be remedied with an online aid. Second, with the GOMS model providing the substantive detail, aiding dialogues could be specified and the interface re-analyzed to determine the impact of online aiding.

Predicting usability problems for online aiding. The most well developed aiding dialogue will be useless if it is provided when the user does not require assistance or instruction. Fortunately, identifying usability problems which may be addressed with online aiding follows directly from a theory-based, task analysis of the user interface. Specifically,

Table 6

Suggested Design Principles for Providing Online Advice Based on the GOMS Model

---

Use goals in online aiding to:

1. Describe what can be done in task-oriented terms (interface actions and objects) for improved initial skill learning.
2. Provide an adjustable level of detail on interface procedures for accommodating the information needs of a wide range of users.
3. Provide procedurally incomplete advice so that users can actively learn for improved long-term performance and understanding with the interface.
4. Provide feedback to users which may help in reminding them of appropriate procedures to use particularly when recovering from errors.
5. Develop modular assistance and instructional dialogues that can be used to describe similar and dissimilar procedural elements of the interface.

Use operators in online aiding to:

1. Describe simple actions, such as pressing specific keys or finding specific objects on the display, that are common to many interface procedures to assist the user in current task performance.
2. Provide detailed knowledge of interface procedures that inexperienced users can actively learn and that more skilled users can combine with other procedural knowledge to improve long-term performance and understanding of the interface.
3. Monitor user actions to provide context sensitive help or to actively diagnose user problems.

Use methods in online aiding to:

1. Present step-by-step interface procedures to assist the user with specific problems.
2. Improve user understanding and acceptance of online advice.
3. Decrease the cognitive load of users who are learning a new interface task by providing an explicit procedure for users to follow.
4. Provide procedural demonstrations of interface procedures so that users can quickly learn simple operations.
5. Map sequences of user's actions to a reduced set of interface goals to help provide context-sensitive advice to users.

Use selection rules in online aiding to:

1. Help users select between multiple interface methods.
2. Provide users with an understanding of representative tasks to increase their knowledge of when to apply specific interface skills.

---

predictions of extreme performance times, learning times, and user memory loads can be described through detailed GOMS models.

Performance times can be estimated with fair accuracy by using the Card, et al. (1983) Keystroke model. With this model, total time to execute a task can be predicted simply as the sum of individual operator times and requires that the interface method be specified at the level of operators (e.g., keystrokes, mouse movements, and glances to and from the display). As a result, if there are alternative interface methods, then the user could be monitored and prompted to use more time efficient methods in specific circumstances. To use the example provided by Finin (1982), if the user's goal was inferred to be "delete the old versions of a file" and the user was repeatedly employing the delete command, then the aiding interface could suggest the purge command since repeated use of delete will yield extreme performance times.

Learning times also can be predicted based on a detailed GOMS analysis formulated as a production rule system (Kieras & Polson, 1985). Using production system models, Polson and Kieras (1985) have found that the number of production rules (i.e., if-then rules) can quantify the amount of procedural knowledge required to learn an interface. These investigators found that a simple count of the number of rules associated with methods and selection rules can predict learning time. Moreover, Polson, Bovair, and Kieras (1987) have found that transfer of training can be predicted by determining the number of new rules to be learned (as opposed to identical rules and rules that can be generalized). Thus, ease of learning can be predicted for a specific interface based on the number of new rules to be learned in a task. If this predicted time is extreme, appropriate training procedures could be implemented to simplify the learning environment (e.g., training wheels interfaces and progressive part training) with similar and dissimilar rules dictating the presentation of interface methods. For example, if a text editor is difficult to learn, a production rule analysis may suggest initial training of procedures for

selecting text since these interface methods are required for many other commands (e.g., copy, delete, etc.).

Finally, the theory-based GOMS model also permits analysis of other cognitive factors such as working memory loads (Kieras & Polson, 1985) which may be an important factor in learning and using a computer interface. Specifically, the memory loads experienced by the user can be estimated by counting the goals and subgoals activated during a cognitive simulation with a production system model. From this analysis, predictions of user errors could be generated with the expectation that periods of high memory loads would result in more errors than periods of low memory loads. High working memory loads also may decrease learning and performance times since the user will have to share their limited capacity working memory when learning interface methods (Kieras & Bovair, 1986). Therefore, if these periods of high working memory loads can be predicted, then additional prompts and cues could be provided by the aiding interface to support error-free and time efficient user performance.

Predicting improvements in usability with online aiding. Once usability problems are identified and initial solutions for aiding the user specified, then the GOMS and the detailed usability prediction models can be used to determine any performance improvements with online aiding (i.e., decreased performance times, learning times, or mental workload). For this to be possible, the aiding dialogues must be specified in enough detail to determine the goals, operators, methods, and selection rules for the user interacting with the aiding interface. However, the approach is feasible if a top-down design strategy is followed using the GOMS model as a method for specifying the aiding dialogues.

The importance of this prediction capability is clear when considering that this review uncovered several aiding dialogues which increased task performance times. Ultimately, this predictive capability could be used as an early indicator of problems with aiding interfaces and

potentially could be used to screen inferior aiding dialogues. Indeed, the intrusiveness of many aiding dialogues may be the result of aiding methods which are inconsistent with the rest of the user interface. Thus, users might have to interact much differently with the online help and tutorials. Specifying the online aiding dialogue and predicting its usability will allow a much more powerful research and design strategy for online aiding. As with other user interfaces, the impact of the aiding interface should be assessed before it is implemented.

CONCLUSIONS

This report has reviewed the literature on aiding interfaces which attempt to assist and instruct users on a computer-based task. The review revealed that many of the efforts to construct interfaces to aid the user have met with mixed success. Many aiding dialogues can only assist the user on the current computer-based task, while other instructional dialogues can only help users acquire knowledge for improved long-term performance. In fact, many aiding interfaces do not improve performance and even when performance enhancements are observed, the characteristics of online aiding which contributed to the improvement often cannot be isolated.

To remedy these problems, a task-analytic approach for research and development of aiding interfaces was described based on the existing GOMS model of human-computer interaction (Card, et al., 1983). This theory-based approach seeks to develop a principled method for the development of online aiding where each component of the GOMS model (goals, operators, methods, and selection rules) provides the framework for the aiding dialogue. In addition, the task-analytic approach permits a quantitative assessment of user interface problems which can be solved with online aiding and also provides a method for predicting any improvements in usability as a result of the aiding dialogues.

Like all formal theoretical models, this approach has limitations which require further research. For example, the GOMS and Keystroke models of Card, et al. (1983) were developed based on skilled and error-free performance. How can these models be applied to aiding users who are generally unskilled and frequently commit errors? Similarly, the GOMS approach does not explicitly address the individual differences of users which may play a role in the success of an aiding interface. The response to these challenges is that these task-analytic procedures approximate user behavior and reveal the interface methods which play a large role in interface usability. Therefore, task-analytic approaches offer few solutions for addressing the individual differences of users. User errors, however, may be handled in a cognitive task analysis by looking at the cognitive processes and interface methods which may lead to errors (e.g., high working memory loads) or by analyzing the error recovery methods of users.

Clearly, theory-based, task-analytic models are a rigorous engineering approach for solving some of the usability problems associated with online aiding interfaces and user interface in general. Cognitive task analyses provide methods to construct aiding interfaces through GOMS models and a way to evaluate the success of the aiding dialogues with usability predictions. Therefore, this approach may be a useful supplement to state-of-the-art iterative design methods which require extensive user testing for the development of aiding interfaces. In fact, this theory-based approach should make it possible to develop systematically aiding interfaces capable of helping computer users on their current task, while also encouraging their continued acquisition of interface skills.

ACKNOWLEDGMENTS

The author would like to thank David E. Kieras for numerous discussions which resulted in the theory-based approach to online aiding and Susan L. Palmiter for her comments on drafts of this report. This work was supported, in part, by a contract from the Office of Naval Research under ONR contract number N00014-87-K-0740 with John J. O'Hare serving as the technical monitor.

REFERENCES

Aaronson, A., & Carroll, J. M. (1987). Intelligent help in a one-shot dialogue: A protocol study. In ***Proceedings of CHI+GI 1987*** (pp. 163-168). New York: ACM.

Al-Awar, J., Chapanis, A., & Ford, W. R. (1981). Tutorials for the first time computer user. ***IEEE Transactions on Professional Communication, 24,*** 30-37.

Anderson, J. R. (1983). ***The architecture of cognition.*** Cambridge, MA: Harvard University Press.

Anderson, J. R. , Boyle, C. F., Farrell, R., & Reiser, B. J. (1984). ***Cognitive principles in the design of computer tutors*** (Tech. Rep. No. ONR-84-1). Pittsburgh, PA: Carnegie Mellon University, Department of Psychology.

Bailey, R. W. (1982). ***Human performance engineering.*** Englewood Cliffs, NJ: Prentice-Hall.

Bauer, D. W., & Eddy, J. K. (1986). The representation of command language syntax. ***Human Factors, 28,*** 1-10.

Black, J. B., Carroll, J. M., & McGuigan, S. M. (1987). What kind of minimal instruction manual is most effective? ***Proceedings of CHI+GI 1987*** (pp. 159-162). New York: ACM.

Borenstein, N. S. (1985). ***The design and evaluation of on-line help systems.*** Unpublished doctoral dissertation, Department of Computer Science, Carnegie-Mellon University, Pittsburgh, PA.

Card, S. K., Moran, T. P., & Newell, A. (1983). ***The psychology of human-computer interaction.*** Hillsdale, NJ: Lawrence Erlbaum.

Carroll, J. M., & Carrithers, C. (1984). Blocking learner errors in a training-wheels system. ***Human Factors, 26,*** 377-390.

Carroll, J. M., & Kay, D. S. (1985). Prompting, feedback, and error correction in the design of a scenario machine. In ***Proceedings of CHI'85: Human Factors in Computing Systems*** (pp. 149-153). New York: ACM.

Carroll, J. M., & Mack, R. L. (1984). Learning to use a word-processor: By doing, by thinking, and by knowing. In J. C. Thomas & M. L. Schneider (Eds.) ***Human factors in computer systems*** (pp. 13-51). Norwood, NJ: Ablex.

Carroll, J. M., Mack, R. L., Lewis, C., Grischkowski, N., & Robertson, S. (1985). Exploring exploring a word processor. ***Human-Computer Interaction, 1,*** 283-307.

Carroll, J. M., & Mazur, S. A. (1986). LisaLearning. ***Computer, 19*** (11), 35-49.

Carroll, J. M., & McKendree, J. (1987). Interface design issues for advice-giving expert systems. ***Communications of the ACM, 30,*** 14-31.

Carroll, J. M., Smith-Kerker, P. L., Ford, J. R., & Mazur, S. A. (1986). ***The minimal manual*** (Research Report RC 11637). Yorktown Heights, NY: IBM T. J. Watson Research Center.

Catrambone, R., & Carroll, J. M. (1987). Learning a word-processing system with training wheels and guided exploration. In ***Proceedings of CHI+GI 1987*** (pp. 169-174). New York: ACM.

Charney, D. H., & Reder, L. M. (1986). Designing interactive tutorials for computer users. ***Human-Computer Interaction, 2,*** 297-317.

Cohill, A. M., & Williges, R. C. (1985). Retrieval of HELP information for novice users of interactive computer systems. ***Human Factors, 27,*** 335-344.

Czaja, S. J., Hammond, K., Blascovich, J. J., & Swede, H. (1986). Learning to use a word-processing system as a function of training strategy. ***Behaviour and Information Technology, 5*** , 203-216.

Digital Equipment Corp. (1986). ***MicroVMS Workstation User's Guide*** (Order Number: AA-EZ24B-TN). Maynard, MA: Digital Equipment Corp.

Dunsmore, H. E. (1980). Designing an interactive facility for non-programmers. In ***Proceedings of the ACM National Computer Conference*** (pp. 475-483). New York: ACM.

Eberts, R., & Brock, J. F. (1984). Computer applications to instruction. In F. A. Muckler (Ed.), ***Human Factors Review: 1984*** (pp. 239-284). Santa Monica, CA: Human Factors Society.

Elkerton, J. (1987). A framework for designing intelligent human-computer dialogues. In G. Salvendy (Ed.), ***Cognitive Engineering in the Design of Human-Computer Interaction and Expert Systems*** (pp. 567-574). Amsterdam, The Netherlands: Elsevier.

Elkerton, J., & Williges, R. C. (1984a). Information retrieval strategies in a file-search environment. ***Human Factors, 26,*** 171-184.

Elkerton, J., & Williges, R. C. (1984b). The effectiveness of a performance-based assistant in an information retrieval environment. In ***Proceedings of the Human Factors Society 28th Annual Meeting*** (pp. 634-638). Santa Monica, CA: Human Factors Society.

Elkerton, J., & Williges, R. C. (1985). A performance profile methodology for implementing assistance and instruction in computer-based tasks. ***International Journal of Man-Machine Studies, 23,*** 135-151.

Elkerton, J., & Williges, R. C. (1987). A summary of experimental research on command-selection aids. In ***Proceedings of INTERACT'87, Second IFIP Conference on Human-Computer Interaction.*** (pp. 937-942). New York: North-Holland.

Elkerton, J., & Williges, R. C. (in press). Dialogue design for intelligent interfaces. In M. H. Chignell, P. A. Hancock, & A. Loewenthal (Eds.), ***Intelligent interfaces: Theory, research, and design.*** New York: North Holland.

Fenchel, R. S. & Estrin, G. (1982). Self-describing systems using integral help. ***IEEE Transactions on Systems, Man, and Cybernetics, 12,*** 162-167.

Finin, T. W. (1982). ***Help advice in task oriented systems*** (Tech. Rep. No. MS-CIS-1982-22). Philadelphia: University of Pennsylvania, The Moore School, Department of Computer and Information Science.

Fischer, G. Lemke, A., & Schwab, T. (1985). Knowledge-based help systems. In ***Proceedings of CHI'85 Human Factors in Computing Systems*** (pp. 161-167). New York: ACM.

Gettys, D. (1986). IF you write documentation, THEN try a decision table. ***IEEE Transactions on Professional Communications, 29,*** 61-64.

Goldstein, I. L. (1987). The relationship of training goals and training systems. In G. Salvendy (Ed.) ***Handbook of human factors*** (pp. 963-975). New York: Wiley.

Grignetti, M. G., Hausmann, C., & Gould, L. (1975). An intelligent on-line assistant and tutor--NLS-SCHOLAR. ***Proceedings of the National Computer Conference, 44,*** 775-781.

Hayes, B. C., & Williges, R. C. (1986). Defining search strategies in information retrieval. In ***Proceedings of the Conference on Systems, Man, and Cybernetics*** (pp. 1092-1096). New York: IEEE

Houghton, R. C. (1984). Online help systems: A conspectus. ***Communications of the ACM, 27,*** 126-133.

Kearsley, G. (1985). Embedded training: The new look of computer-based instruction. ***Machine-Mediated Learning, 1,*** 279-296.

Kieras, D. E., & Bovair, S. (1986). The acquisition of procedures from text: A production-system analysis of transfer of training. ***Journal of Memory and Language, 25,*** 507-524.

Kieras, D. E., & Polson, P. G. (1985). An approach to the formal analysis of user complexity. ***International Journal of Man-Machine Studies, 22,*** 365-394.

Kyllonen, P. C., & Alluisi, E. A. (1987). Learning and forgetting facts and skills. In G. Salvendy (Ed.) ***Handbook of human factors*** (pp. 124-153). New York: Wiley.

Lewis, C., Casner, S., Schoenberg, V., & Blake, M. (1987). Analysis-based learning in human-computer interaction. ***Proceedings of INTERACT'87, Second IFIP Conference on Human-Computer Interaction.*** (pp. 275-280). New York: North-Holland.

Magers, C. S. (1983). An experimental evaluation of on-line HELP for non-programmers. In ***Proceedings of CHI'83: Human Factors in Computing Systems*** (pp. 277-281). New York: ACM.

Mason, M. V. (1986). Adaptive command prompting in an online documentation system. ***International Journal of Man-Machine Studies, 25,*** 33-51.

McKendree, J., & Carroll, J. M. (1987). Impact of feedback content in initial learning of an office system. In ***Proceedings of INTERACT'87, Second IFIP Conference on Human-Computer Interaction*** (pp. 855-859). New York: North-Holland.

Meister, D. (1985). ***Behavioral analysis and measurement methods.*** New York: Wiley.

Orwick, P., Jaynes, J. T., Barstow, T. R., & Bohn, L. S. (1986). DOMAIN/DELPHI: Retrieving documents online. In ***Proceedings of CHI'86 Human Factors in Computing Systems*** (pp. 114-121). New York: ACM.

Owen, D. (1986). Answers first, then questions. In D. A. Norman & S. W. Draper (Eds.), ***User centered system design*** (pp. 361-376). Hillsdale, NJ: Lawrence Erlbaum.

Polson, P. G., Bovair, S., & Kieras, D. E. (1987). Transfer between text-editors. In ***Proceedings of CHI+GI 1987*** (pp. 27-32). New York: ACM.

Polson, P. G., & Kieras, D. E. (1985). A quantitative model of the learning and performance of text-editing knowledge. In ***Proceedings of CHI'85: Human Factors in Computing Systems*** (pp. 207-212). New York: ACM.

Reder, L. M., Charney, D. H., & Morgan, K. (1986). The role of elaborations in learning a skill from an instructional text. ***Memory and Cognition, 14,*** 64-78.

Relles, N. (1979). ***The design and implementation of user-oriented systems.*** Unpublished doctoral dissertation, University of Wisconsin, Madison, WI.

Riley, M. S., & O'Malley, C. (1984). Planning nets: A framework for analyzing user-computer interactions. In ***Proceedings of INTERACT'84, First IFIP Conference on Human-Computer Interaction*** (pp. 513-518). New York: Elsevier.

Robinson, E. R. N., & Knirk, F. G. (1984). Interfacing learning strategies and instructional strategies in computer training programs. In F. A. Muckler (Ed.), ***Human Factors Review: 1984*** (pp. 209-238). Santa Monica, CA: Human Factors Society.

Sebrechts, M. M., Deck, J. G., & Black, J. B. (1983). A diagrammatic approach to computer instruction for the naive user. ***Behavior Research Methods and Instrumentation, 15,*** 200-207.

Shneiderman, B. (1986). ***Designing the user interface.*** Reading, MA: Addison-Wesley.

Sleeman, D., & Brown, J. S. (1982). ***Intelligent tutoring systems.*** New York: Academic Press.

Sondheimer, N. K., & Relles, N. (1982). Human factors and user assistance in interactive computer systems: An introduction. ***IEEE Transactions on Systems, Man, and Cybernetics, 12,*** 102-107.

Swezey, R. W. (1987). Design of job aids and procedure writing. In G. Salvendy (Ed.) ***Handbook of human factors*** (pp. 1039-1057). New York: Wiley.

Teitelman, W. (1985). Cedar programming environment. ***SIGGRAPH Video Review, 19,*** 8.

University of Michigan Computer Center. (1984). ***MTS: The Michigan Terminal System.*** Ann Arbor, MI: The University of Michigan.

Walker, J. (1985). The document examiner. ***SIGGRAPH Video Review, 19,*** 4.

Wilensky, R. , Arens, Y., & Chen, D. (1984). Talking to UNIX in English: An overview of UC. ***Communications of the ACM, 27,*** 574-593.

Williges, R. C., Williges, B. H., & Elkerton, J. (1987). Software interface design. In G. Salvendy (Ed.) ***Handbook of human factors*** (pp. 1416-1449). New York: Wiley.

Xerox Corp. (1986). ***Training and reference: Xerox ViewPoint*** (Publication No: 610E01460). El Segundo, CA: Xerox Corp.

Distribution List for This Report

OSD

Dr. Earl Alluisi
Office of the Deputy Under Secretary
of Defense
OUSDRE (E & LS)
Pentagon, Room 3D129
Washington, DC 20301

DEPARTMENT OF THE NAVY

Aircrew Systems Branch
Systems Engineering Test
Directorate
U.S. Naval Test Center
Patuxent River, MD 20670

Dr. Glen Allgaier
Artificial Intelligence Branch
Code 444
Naval Electronics Ocean System Center
San Diego, CA 92152b1

Dr. L. Chmura
Computer Sciences & Systems
Code 5592
Naval Research Laboratory
Washington, DC 203650

Commander
Naval Air Systems Command
Crew Station Design
NAVAIR 5313
Washington, DC 20361

Director
Technical Information Division
Code 2627
Naval Research Laboratory
Washington, DC 20375-5000

Dr. Eugene E. Gloye
ONR Detachment
1030 East Green Street
Pasadena, CA 91106-2485

Human Factors Branch
Code 3152
Naval Weapons Center
China Lake, CA 93555

Mr. Philip Andrews
Naval Sea Systems Command
NAVSEA 61R2
Washington, DC 20362

Dr. Stanley Collyer
Office of Naval Technology
Code 222
800 North Quincy Street
Arlington, VA 22217-5000

Dean of the Academic Departments
U.S. Naval Academy
Annapolis, MD 21402

Dr. Robert A. Fleming
Human Factors Support Group
Naval Personnel Research &
Development Center
1411 South Fern Street
Arlington, VA 22202-2896

Mr. Jeff Grossman
Human Factors Laboratory, Code 71
Navy Personnel R&D Center
San Diego, CA 92152-6800

Human Factors Department
Code N-71
Naval Training Systems Center
Orlando, FL 32813

Human Factors Engineering
Code 441
Naval Ocean Systems Center
San Diego, CA 92152

Dr. Michael Letsky
Office of the Chief of Naval
Operations (OP-01B7)
Washington, DC 20350

LCDR Thomas Mitchell
Code 55
Naval Postgraduate School
Monterey, CA 93940

Capt. W. Moroney
Naval Air Development Center
Code 602
Warminster, PA 18974

CDR James Offutt
Office of the Secretary of Defense
Strategic Defense Initiative Organization
Washington, DC 20301-5000

Dr. Randall P. Schumaker
NRL A.I. Center
Code 7510
Naval Research Laboratory
Washington, DC 20375-5000

Dr. A.L. Slafkosky
Scientific Advisor
Commandant of the Marine Corps
Washington, DC 20380

Mr. H. Talkington
Engineering & Computer Science
Code 09
Naval Ocean Systems Center
San Diego, CA 92152

Capt. Thomas Jones
Code 125
Office of Naval Research
800 North Quincy Street
Arlington, VA 22217-5000

Lt. Dennis Mc Bride
Human Factors Branch
Pacific Missile Test Center
Point Mugu, CA 93042

Dr. George Moeller
Human Factors Department
Naval Submarine Medical Res Lab
Naval Submarine Base
Groton, CT 06340-5900

Dr. A.F. Norcio
Computer Sciences & Systems
Code 5592
Naval Research Laboratory
Washington, DC 20375-5000

Perceptual Science Program (3 copies)
Office of Naval Research
Code 1142PS
800 North Quincy Street
Arlington, VA 22217-5000

LCDR T. Singer
Human Factors Engineering Division
Naval Air Development Center
Warminster, PA 18974

Special Assistant for Marine
Corps Matters
Code OOMC
Office of Naval Research
800 North Quincy Street
Arlington, VA 22217-5000

DEPARTMENT OF THE ARMY

Director, Organizations and Systems
Research Laboratory
U.S. Army Research Institute
5001 Eisenhower Avenue
Alexandria, VA 22333-5600

Dr. Michael Drillings
Basic Research Office
Army Research Institute
5001 Eisenhower Avenue
Alexandria, VA 22333-5600

Technical Director
U.S. Army Human Engineering Laboratory
Aberdeen Proving Ground, MD 21005

Dr. Edgar M. Johnson
Technical Director
U.S. Army Research Institute
Alexandria, VA 22333-5600

DEPARTMENT OF THE AIR FORCE

Dr. Kenneth R. Boff
AF AMRL/HE
Wright-Patterson AFB, OH 45433

Mr. Yale Smith
Rome Air Development
Center, RADC/COAD
Griffiss AFB
New York 13441-5700

Dr. Charles Bates, Director
Human Engineering Division
HSAF AMRL/HES
Wright-Patterson AFB, OH 45433

OTHER GOVERNMENT AGENCIES

Defense Technical Information
Center
Cameron Station, Bldg. 5
Alexandria, VA 22314 (2 COPIES)

Dr. M.C. Montemerlo
Information Sciences &
Human Factors Code RC
NASA HQS
Washington, DC 20546

Dr. Clinton Kelly
Defense Advanced Research
Projects Agency
1400 Wilson Boulevard
Arlington, VA 22209

OTHER ORGANIZATIONS

Dr. H.E. Bamford
Program Director
Division of Information,
Robotics and Intelligent Systems
National Science Foundation
Washington, DC 20550

Dr. Michael Athans
Massachusetts Inst. of Technology
Lab Information & Decision Systems
Cambridge, MA 02139

Ms. Bonnie E. John
Department of Psychology
Carnegie-Mellon University
Pittsburgh, PA 15213

Dr. Thomas G. Moher
Dept. of Electrical Engineering
and Computer Science
University of Illinois at Chicago
P.O. Box 4348
Chicago, IL 60680

Dr. Jesse Orlansky
Institute for Defense Analyses
1801 N. Beauregard Street
Alexandria, VA 22311

Dr. Scott Robertson
Department of Psychology
Rutgers University
Busch Campus
New Brunswick, NJ 008903

Dr. Allen Newell
Department of Computer Science
Carnegie-Mellon University
Pittsburgh, PA 15213

Dr. Richard Pew
Bolt Beranek & Newman, Inc.
10 Moulton Street
Cambridge, MA 02238

Dr. H.P. VanCott
NAS-National Research Council
(COHF)
2101 Constitution Avenue, NW
Washington, DC 20418

END
DATE
FILMED
3-1988
DTIC